LAMP-TR-001
CFAR-TR-839
CS-TR-3688

September 1996

Compressed Video Segmentation

Vikrant Kobla, David Doermann, Azriel Rosenfeld

Language and Media Processing Labratory
Instititue for Advanced Computer Studies
College Park, MD 20742

Abstract

Segmentation of video into shots and scenes in the compressed domain allows rapid, real-time analysis of video content using standard hardware. This paper presents robust techniques for parsing MPEG-compressed video sequences into shots based on their physical structure and further into scenes based on their semantic structure by identifying changes in content and camera motion. The analysis is performed in the compressed domain using available macroblock and motion vector information, and if necessary, DCT information. Motion vector analysis yields a qualitative description of the camera motion and is used to subdivide shots into subshots. Key frames for the shots and scenes can be used for browsing, indexing, and retrieval.

***The support of the LAMP Technical Report Series and the partial support of this research by the National Science Foundation under grant EIA0130422 and the Department of Defense under contract MDA9049-C6-1250 is gratefully acknowledged.

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE
SEP 1996

2. REPORT TYPE

3. DATES COVERED
00-09-1996 to 00-09-1996

4. TITLE AND SUBTITLE
Compressed Video Segmentation

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Language and Media Processing Laboratory,Institute for Advanced Computer Studies,University of Maryland,College Park,MD,20742-3275

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:

a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
unclassified	**unclassified**	**unclassified**		**26**	

1 Introduction

Automated video parsing involves dividing a video sequence into shots and scenes, and selecting one or more key frames to represent each shot. These key frames can then be used for browsing, indexing, retrieval and other applications which operate on a single shot. Temporal and content relationships between key frames can then be used to characterize the structure of the video. Due to the presence of a considerable amount of camera motion or object motion in some clips, there may be very little similarity between, for example, frames early in a sequence and frames later in the same shot. In these cases, more than one key frame may be required.

When segmenting video, it is necessary to consider both physical and semantic partitioning. Physical partitions result from stopping and restarting the camera or artificially combining two sequences using fades, dissolves or other cinematic effects. Semantic partitions are typically content-related and depend to some extent on the application. For example, as a camera pans over a scene, the entire sequence may be contained in a single shot, but the content of the scene may change to such an extent that it should be considered as a new semantic entity. Our discussion is based on a dichotomy between the physical structure, which divides the sequence into “shots”, and the semantic structure, which divides it into “scenes”. Subdivisions of shots based on other physical criteria such as changes in camera motion are termed “subshots”.

The relationships between shot changes and scene changes may be somewhat fuzzy. Typically, a shot change will imply a scene change. For example, when a sequence shows one person and then changes to another person, a scene change has clearly occurred. If, however, two shots show different views of the same person or place, it is not obvious that a scene change should be identified. Similarly, if a camera pan is used to change the view from one person or place to another, the shot change criteria may not be satisfied, but the scene change criteria may. For consistency and simplicity, we will assume that a scene never crosses a shot boundary and that any inconsistencies in the relationships between scenes can be resolved at a higher level.

A shot in a video clip can thus be defined as a maximal sequence of frames resulting from a continuous, uninterrupted recording by the input device, typically a camera. Parsing involves determining which shot each frame in the sequence (or clip) belongs to. The first step in parsing is determining where shot changes occur in the clip, and the second step is to refine the segmentation into subshots which exhibit consistent physical properties such as camera motion. Clues obtained from the subshots are further used to subdivide the shots into scenes which exhibit visually consistent content. For example, camera motion subshots may predict the need for subdivision into scenes which have substantially different content in different parts of the continuous sequence of frames. We can define other content-based “scene change” criteria for this subdivision, such as objects appearing or disappearing, variations in illumination, or other application-defined events. In the case of camera motion, the subshot results can also be used to refine the process by correcting false detections by the shot change detector. Finally, using shot change, subshot, and scene information, key frames are extracted from the sequences and subsequences. In this paper we are concerned primarily with the detection of shot changes, camera motion[1], and the selection of key frames.

Although a lot of work has been done on compressed-domain video analysis and shot change detection, many of the algorithms fail in certain troublesome cases which will be described in detail later. This paper presents a robust algorithm that performs well for MPEG video encoded with any pattern of I, P, and B frames by identifying these troublesome cases, and applying further analysis only to these cases to confirm the presence of a shot change.

Section 2 of this paper gives some background on compressed domain analysis. Section 2.1 describes the MPEG representation and the components of the compressed stream that are meaningful for shot change analysis, Section 2.2 describes related work on the subject, and Section 2.3 gives a brief overview of our system. Section 3 and its subsections describe our algorithm for shot change detection, and Section 4 describes the motion vector analysis techniques used for identifying camera motion. Section 5 explains the shot subdivision concept and key frame identification. A summary of the experiments, results and a discussion of their implications are presented in Section 6.

2 Compressed Domain Analysis

The problem of segmentation and key frame extraction is important in early stages of video analysis because of the large amount of compressed data that needs to be processed in order to reduce the apparent redundancy, while at the same time keeping enough data to yield a concise summary of the video. The analysis of compressed video can proceed in one of two fundamental ways. The first is by decompressing some or all of the video and using the individual frames to gather shot change information (e.g. using color histograms) and motion information in the pixel domain. The second involves exploiting encoded information contained in the compressed representation and does not require the overhead of decompression. This paper deals with the parsing of video using the information available in MPEG-compressed video.

2.1 MPEG Compression

The Moving Picture Expert Group (MPEG) standard for digital video is arguably the most widely accepted international video compression standard. The MPEG encoding algorithm [6] relies on two basic techniques: block-based motion compensation to capture temporal redundancy, and transform-domain-based compression to capture spatial redundancy. Motion compensation techniques are applied using both

[1] Throughout this paper, we assume that "camera motion" refers to operations like panning, tilting, and zooming a fixed camera and not to actual change in position of the camera.

predictive and interpolative techniques. The prediction error signal is further compressed using spatial redundancy reduction techniques. The fact that temporal and spatial changes are fundamental for segmentation makes MPEG an ideal candidate for compressed domain analysis.

An MPEG stream [5] consists of three types of frames — I,P, and B frames — occurring in a repetitive pattern. An I frame is an anchor frame that is simply a JPEG encoding [14] of its corresponding pixel image. A P frame is predicted from its preceding I or P frame, and a B frame is predicted from both its preceding and following I and/or P frames. Figure 1 shows the predictive relationships. All three types of frames are ultimately encoded using 2D DCT, quantization, and run-length coding. In the I frames, the DCT information is derived directly from the image samples, but in the P and B frames, the DCT information is derived from the residual error after prediction. The motion vector information is also differentially coded. Each frame in the clip is divided into blocks of 16×16 pixels called Macroblocks (MBs), and most of the low-level processing, including spatial and temporal redundancy reduction, is performed at this level of spatial resolution.

During the encoding process, a test is made on each macroblock in a P frame, to see if that macroblock can be predicted from its corresponding block in the previous I/P frame with a possible offset to compensate for motion. If it appears that it would be more expensive to predict and encode it than to code it directly (using JPEG techniques), then that block is not predicted but is intra-coded. This typically occurs if the current macroblock does not have much in common with the previous one. Every P frame therefore consists of intra-coded MBs and forward-predicted MBs. There are also skipped MBs which resemble forward-predicted MBs, since they are identical to some 16×16 block of the previous reference frame.

Similarly, for each macroblock in a B frame, a test is made to see if it can be predicted from both its previous and next I/P frames, its previous I/P frame alone, its next I/P frame alone, or if it cannot be predicted from any reference frame and thus has to be intra-coded. A MB can also be skipped if it is identical to some part of the reference frames and the residual error becomes zero. Each skipped MB behaves identically to the previous non-skipped MB in the current slice of the current frame.[2] In other words, the first MB of a slice cannot be skipped, and if the MB type and motion vectors of this non-skipped MB are sufficient for any continuous string of MBs in the slice immediately following the non-skipped MB, those MBs are skipped. Each MB in a B frame can be of any one of these five types.

The MPEG standard does not specify which techniques are to be used for motion estimation and prediction during the encoding process. Thus, little can be assumed about the quality of the prediction obtained while encoding. Nevertheless, we assume that the prediction techniques are reasonable enough to yield reliable motion vector and macroblock data. The reader should refer to two articles by Le Gall [5, 6] for more information on MPEG.

[2] MBs in a single frame are grouped into slices of MBs, as a means of layering.

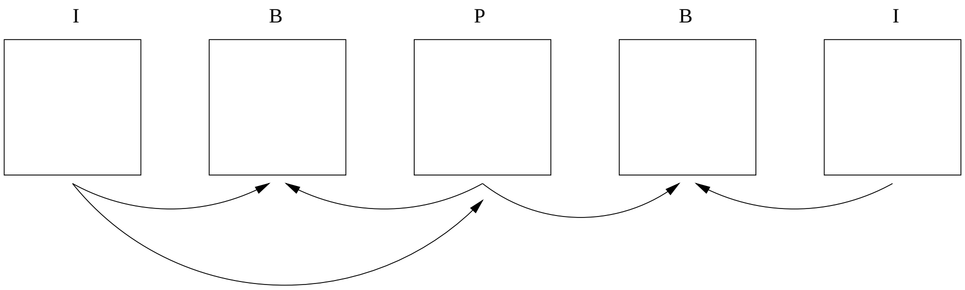


Figure 1: Predictive relationships between I, P and B frames

The computationally expensive step in decompressing MPEG video is the inverse DCT (IDCT), which should be avoided if possible. The information available in the representation without performing IDCT includes the type of each MB, the DCT coefficients of each block, and the motion vector components for the forward, backward, and bidirectionally predicted MBs. The approach followed in this paper involves utilizing all three: the MB types, DCT coefficients, and motion vectors.

2.2 Related Work

The problem of shot change detection in uncompressed video sequences has been addressed by many researchers and a number of papers on the subject have appeared in the literature [2, 4, 10, 12, 17, 18]. In general, the techniques described in these papers are of good quality as they operate at the lowest level, the pixels.

Nagasaka and Tanaka [10] describe a detailed study of seven different evaluation functions that work on the gray-level and color signals in the pixel domain. Zhang et al. [18] describe three techniques, which include pairwise comparison of pixels, likelihood ratio, and histogram of intensity levels as metrics for detecting cuts in a video sequence. They also describe a technique for detecting gradual transitions, and use motion analysis to remove false detections due to camera movements. Hampapur et al. [4] use a model-based top-down approach to design feature detectors for various edit effects. Shahraray [12] employs block-matching techniques to filter out motion and then measures the similarity/disparity between frames, to detect abrupt and gradual shot changes. Cherfaoui and Bertin [2] describe a technique based on camera movements. Zabih et al. [17] describe a technique using intensity edges that detects various edits such as cuts, dissolves, fades, etc.

A lot of work has also been done on detecting shot changes in the compressed domain [1, 7, 9, 11, 13, 16, 19]. The earliest approaches were by Arman et al. [1], who used only the DCT information of I-frames to detect disparity. This technique works well with clips containing only I-frames (XING format or Motion JPEG), but has problems if the clip contains P and B frames and the I-frames are spaced widely apart. Due to the reduction in temporal resolution, there may be false detections due to large motions in the scene. If there is more than one shot change within two

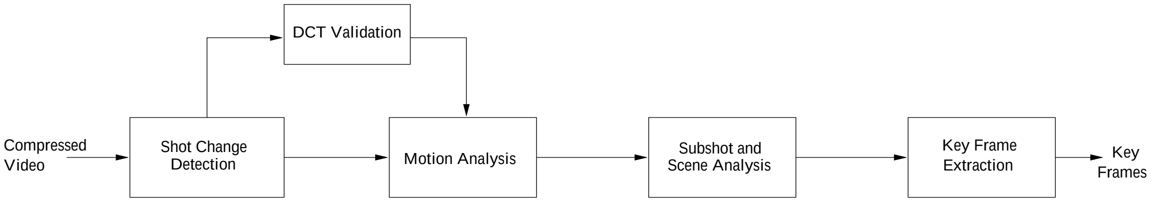


Figure 2: Flow diagram of our system.

consecutive I-frames, these techniques cannot detect all of them nor can they pinpoint where the transition occurred. Sethi and Patel [11] also used only I-frames and thus their approach suffered from similar problems.

In cases where P and B frames exist in a video clip, there are techniques that allow the actual DCT values of the pixel data of these frames to be calculated or estimated using the DCT values of the reference frames and the DCT values of the residual error without computing the IDCT. There has been recent progress in this area of estimating DCT values of arbitrary blocks from component DCT blocks [15]. These techniques would allow the P and B frames in an MPEG clip to be treated just like any other I frame and there would be no loss of temporal resolution, as noted above. This technique was employed by Yeo and Liu [16] and by Shen and Delp [13]. Zhang et al. [19] used a hybrid approach involving a DCT comparison technique for the I-frames and a motion vector validity scheme for the P and B frames. Meng et al. [9] also used the validity of motion vectors to detect shot changes. They described an adaptive technique for determining the thresholds used in their algorithm. A large percentage of the shot changes can be easily detected by performing an analysis of the macroblock information only. This is faster than using DCT information. This technique was also used by Liu and Zick [7].

The algorithm described in this paper uses the MB type information to detect shot changes. For the cases where MB type information alone is insufficient to confirm the existence of a shot change, further DCT analysis is performed. The algorithm is thus very fast as it uses only the MB information for the most part and needs to revert to DCT analysis only for certain rare cases. These special cases were not considered by researchers who used solely the MB type data.

2.3 System Overview

Figure 2 shows the flow diagram of the segmentation and key frame extraction components of our system. Compressed video in MPEG format is first segmented into shots and for certain cases, DCT validation is performed. The segmented video is then passed on to the motion vector analysis routine where global camera motion information is extracted. Using the camera motion information, shots are subdivided into subshots. Finally, key frames are extracted from these shots and subshots.

3 Shot Change Detection

Shot change detection involves identifying the frame(s) where a transition takes place from one shot to another. A change which occurs exactly between two frames is called a *cut* or a *break*. Examples of transitions that occur gradually over several frames include fades, dissolves, wipes, and other special effect edits. This section focuses primarily on determining where breaks occur, since they are the most common type of shot change encountered.

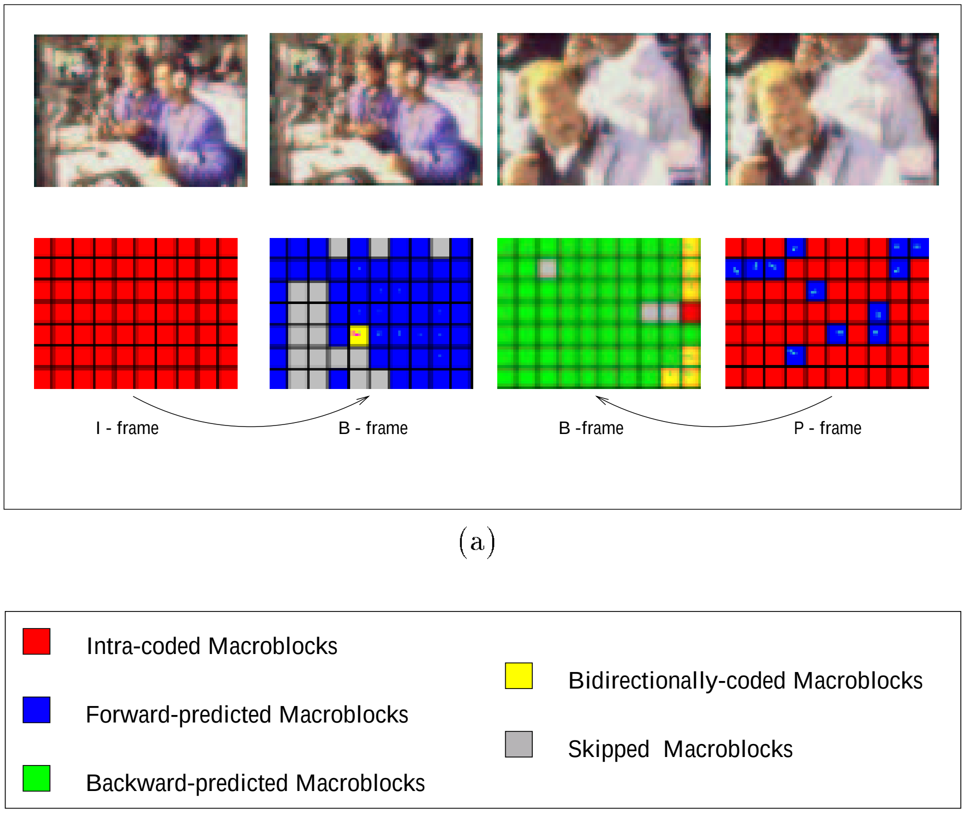
3.1 Cut Detection Using Macroblock Types

Segmentation using macroblock information exploits the inter-frame coherence that exists in a video clip. This technique can be applied to any type of MPEG sequence consisting of I, P and B frames except XING. A test can be made to see if the MPEG file is in XING format, and if it is, any algorithm employing pure DCT information can be used to detect the shot changes [1, 11].

As mentioned earlier, an MPEG stream contains I,P and B frames. If a P frame contains primarily intra-coded MBs, this suggests that these MBs could not be predicted from the previous reference frame, and that there is a high probability of a shot change having occurred somewhere between the previous I/P frame and the current P frame.

If in a B frame, a majority of the MBs are forward-predicted from the previous I/P frame, then there is a high probability that a shot change occurs between the current frame and the next I/P frame. If a majority of the MBs in a B frame are backward-predicted, then the probability of a shot change having occurred between the previous I/P frame and the current frame is high. Within any two reference frames, the B frames on either side of a shot change must behave according to these observations. That is, for all B frames prior to the shot change, the first observation must hold true, and for all B frames after the shot change, the second observation must hold true. The tests for the presence of these features are employed in the detection process.

Figure 3 shows a set of four consecutive frames that include a shot change. Their corresponding macroblock display is shown beneath each frame. The first frame is an I-frame that is used for forward prediction for the last P-frame. Since a shot change occurred between these frames, this was not possible, as is evident from the macroblock display in which the P-frame contains primarily intra-coded MBs. These two frames act as reference frames for the two intermediate B-frames. The first B-frame could only be forward predicted from the I-frame and the second B-frame could only be backward predicted from the P-frame, due to the cut occurring between them.


I - frame
B - frame
B -frame
P - frame
🟥 Intra-coded Macroblocks
🟦 Forward-predicted Macroblocks
🟩 Backward-predicted Macroblocks
🟨 Bidirectionally-coded Macroblocks
⬜ Skipped Macroblocks

(a)

(b)

Figure 3: Shot Change: (a) A set of four consecutive frames, their associated macroblocks and motion vectors. (b) The macroblock encoding legend.

3.1.1 Approach

The algorithm counts the number of each type of MB in each frame. A Resemblance Value (RV) is computed that measures the similarity of each frame to its reference frame. This value acts as a metric that is used directly to detect breaks in the sequence. For P frames, the RV is the sum of the number of forward-predicted MBs and the number of skipped MBs. Thus, if the RV of a P frame is very small, it would imply that it has little resemblance to its previous reference frame. For the B frames, it is the sum of the number of bidirectionally-predicted MBs, the number of skipped MBs that are bidirectionally predicted, and the minimum of the number of forward-predicted (skipped and non-skipped) and backward-predicted (skipped and non-skipped) MBs. Thus, if the RV of a B frame is small, it would imply that the two reference frames that this B frame was interpolated from have little resemblance to one another. Once the RVs for the frames have been determined, each pair of reference frames is considered sequentially and tests are made to see if a break lies at or within those two frames.

3.1.2 Macroblock-based Cut Detection Algorithm

Let us define some terminology.

N - Total number of MBs in a frame.

N_f - Number of forward-predicted MBs.

N_b - Number of backward-predicted MBs.

N_{fb} - Number of bidirectionally-predicted MBs.

N_s - Number of skipped MBs.

N_{sf} - Number of skipped MBs that are actually forward-predicted.

N_{sb} - Number of skipped MBs that are actually backward-predicted.

N_{sfb} - Number of skipped MBs that are actually bidirectionally-predicted.

A discussion of the thresholds $t_1 - t_6$ that are used and how they can be derived is presented in Section 6. We begin by examining the following relationships:

$$N = N_f + N_b + N_{fb} + N_s$$

$$N_s = N_{sf} + N_{sb} + N_{sfb}$$

Let $\mathcal{I}$ be the set of I frames, $\mathcal{P}$ the set of P frames, and $\mathcal{B}$ the set of B frames in the clip. A Resemblance Value RV_i is assigned to each frame i in a video clip as follows [3]:

[3] The RV for an I-frame is not used in the analysis.

$$RV_i = \begin{cases} 0 & \text{if} \quad i = 0 \\ RV_{i-1} & \text{if} \quad i \in \mathcal{I} \quad \& \quad i \neq 0 \\ N_f + N_s & \text{if} \quad i \in \mathcal{P} \\ N_{fb} + N_{sfb} + \min(N_f + N_{sf}, N_b + N_{sb}) & \text{if} \quad i \in \mathcal{B} \end{cases}$$

For clips containing only I and P frames: If the video clip contains only I and P frames, then a new sequence is declared to begin at frame i, $i \in \mathcal{P}$, if

$$\frac{RV_i}{N} < t_1,$$

where t_1 is a threshold. A DCT analysis of consecutive I frames is performed to identify shot changes occurring just before an I frame. If a DCT analysis indicates the presence of a shot change between consecutive I frames and the P frames between them are well predicted, then a new sequence is declared to begin at the latter I frame. If P frames are present in a clip, B frames are typically also present, which we discuss next.

For clips containing B frames: If the video clip also contains B frames, then let $\mathcal{R}$ be the set of reference frames:

$$\mathcal{R} = \mathcal{I} \cup \mathcal{P}$$

We consider two consecutive reference frames, R_i and R_j. Define *domain* $\mathcal{D}$ to be the set containing all B frames between any two reference frames R_i and R_j, and the latter reference frame R_j.

$$R_i \underbrace{BB...BR_j}_{domain\ \mathcal{D}}$$

For any frame $k \in \mathcal{D}$, if the four conditions listed below are satisfied, then a new sequence is declared to begin at frame k.

Condition 1 : If $R_j \in \mathcal{P}$, then

$$\frac{RV_j}{N} < t_2$$

for threshold t_2. This condition is used in a screening test. If this fails, tests for the remaining conditions can be omitted since this P frame is predicted well from the previous reference frame, suggesting that no cut is present in this domain.

Condition 2 :

$$\frac{RV_l}{N} < t_3 \quad \forall l \in \mathcal{D}$$

where t_3 is a threshold. This tests to see if all frames in the domain behave consistently in suggesting the little resemblance between the two reference frames across the B frames.

Condition 3 : For all B frames before k in $\mathcal{D}$ (if any),

$$N_f - N_b > N_{fb}$$

If this test is passed, it suggests that all these B frames are predicted primarily from their previous reference frame.

Condition 4 : For all B frames after and including k in $\mathcal{D}$ (if any),

$$N_b - N_f > N_{fb}$$

If this test is passed, it suggests that all these B frames are predicted primarily from their next reference frame.

Figure 4 shows the histogram of RVs for a clip. All the valleys that fall below the threshold indicated by the dotted line correspond to detected cuts, except for the one shown by the arrow, which is not detected because all the conditions are not satisfied. In particular, condition 2 is not satisfied, as the IPB pattern for the clip was repetitive with IBBPBB, there are three frames in each domain, and only one RV dips below the threshold.

3.2 DCT Validation

DCT validation is used to confirm the existence of shot changes for which the macroblock information is found to be insufficient.

An analysis of the macroblock types alone does not always provide sufficient information to indicate that a shot change occurs between two frames. This typically occurs when there is little motion around the shot change, and a majority of the MBs are skipped. Similarly, a large number of MBs are skipped if coarse quantization scales are used during the encoding process. Skipped MBs do not provide much information about the underlying pixels except for the fact that they are identical to some part of one of the reference frames. For example, if a skipped MB is identical to the corresponding MB in the previous reference frame, it provides no information as to its relation to the corresponding MB in the next reference frame. Instead of being skipped, if it had been bidirectionally predicted, it would have enhanced the belief that a shot change does not exist within the closed domain of the two reference frames. In such cases, performing an analysis of the DCT data encoded in the MPEG stream helps to ascertain the existence of cuts in the video clip.

Figure 5 shows an example of a situation in which a shot change occurs over a sequence of B frames in which the last frame of the previous sequence is substantially

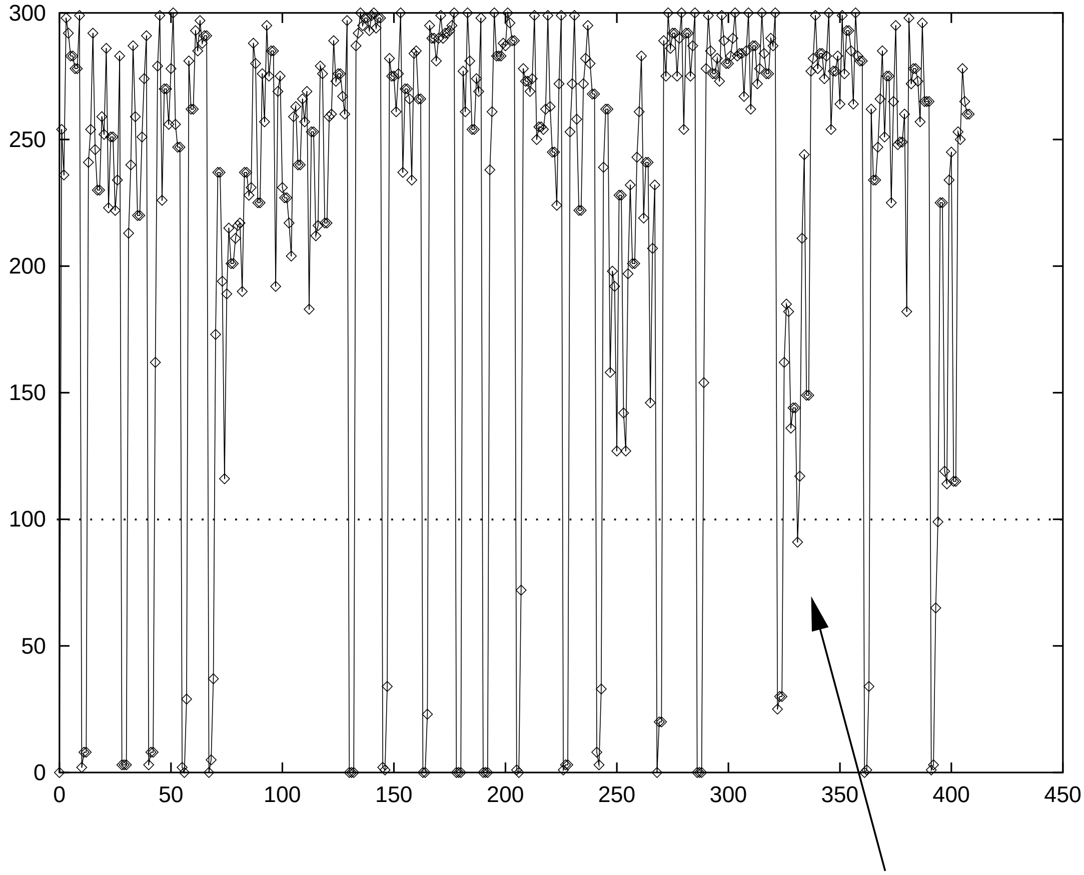


Figure 4: Histogram of RV values. The valley indicated by the arrow produces a valid hypothesis, but is not confirmed because all the conditions are not satisfied.

similar to the previous P frame. The new sequence begins at a B frame that is identical for the most part with the next I frame. Thus in both of the B frames, a majority of the MBs are skipped.

Figure 6 shows an example where no shot change occurs between the two B frames which behave in a manner similar to that described in the example of Figure 5. This confusion occurs only when there is no predictive reference after these B frames to validate the shot change.

Our algorithm pinpoints these troublesome cases and uses the DCT data of the two I frames that straddle the B frames to confirm the existence of a shot change. The analysis involves counting the number of DC[4] coefficients of corresponding blocks that differ in value by more than a given threshold, t_5. If this count exceeds another threshold value, t_6, then a shot change could possibly have taken place between the two frames. In the case of Motion-JPEG, this turns out to be a fairly robust

[4] Of the 64 DCT coefficients, the coefficient with zero frequency in both dimensions is called the 'DC coefficient', while the remaining 63 are called the 'AC coefficients'.

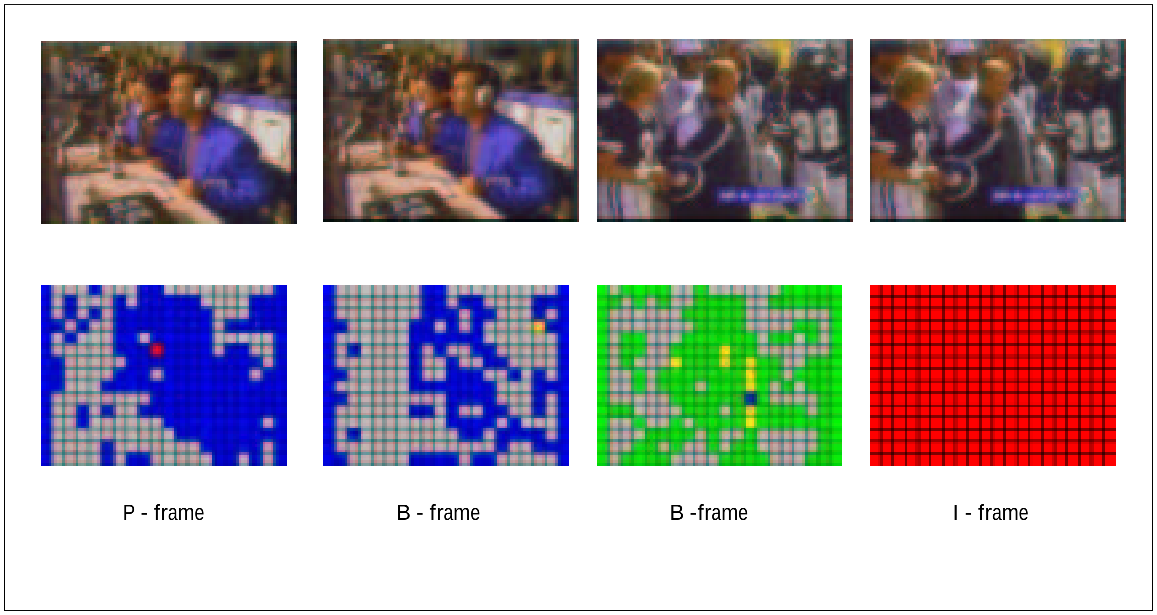


Figure 5: Example with shot change having an ambiguous MB pattern. See legend in Figure 3.

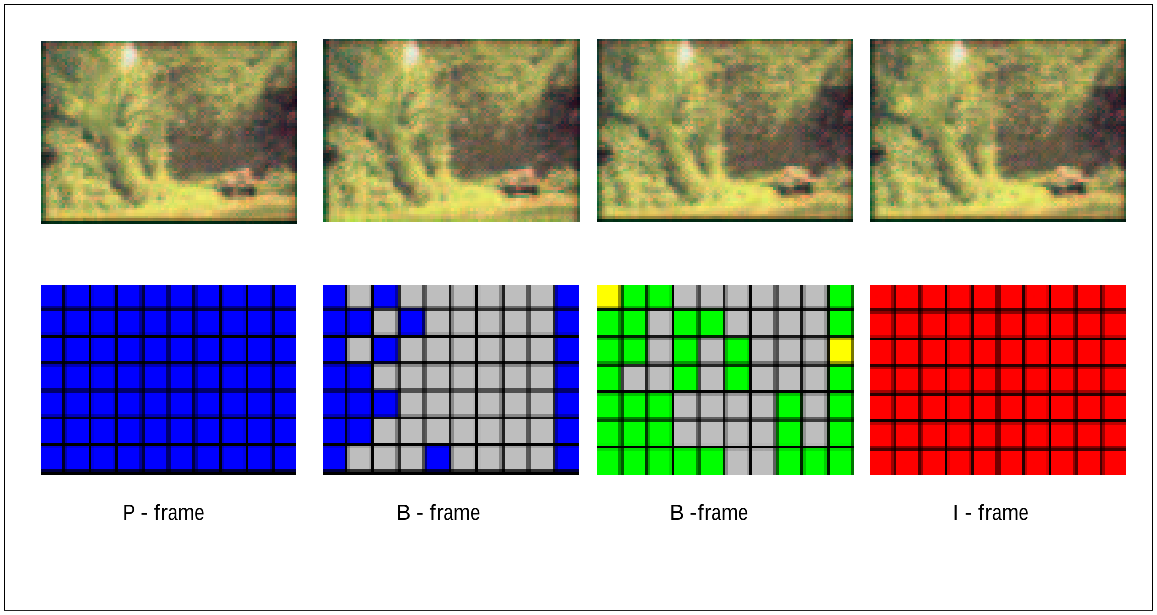


Figure 6: Example without a shot change having an ambiguous MB pattern. See legend in Figure 3.

mechanism for video segmentation.

3.2.1 Algorithm

For any domain $\mathcal{D}$, if a shot change is detected in that domain by the above procedure, and if the reference frame at the end of the domain is an I-frame, then if for any B frame around the cut in $\mathcal{D}$,

$$\frac{N_s}{N} > t_4,$$

where t_4 is a derived threshold, we do a DCT analysis for the domain as follows:

Let I_n be the I-frame in D, and let I_m be the previous I-frame. Define

$$\delta_{ij} = \begin{cases} 1 & \text{if} \quad | DCT(I_n)^0_{ij} - DCT(I_m)^0_{ij} | > t_5 \\ 0 & \text{otherwise} \end{cases}$$

re DCT() is a function that takes the DCT of the given frame. $DCT(I_m)^0_{ij}$ is the DC coefficient of the I_m frame. A shot change is validated if

$$\frac{\sum_{ij} \delta_{ij}}{N} > t_6$$

Any other similarity metric using the DCT coefficients would also suffice. The method used here was simple. If the shot change needs to be rejected, the fact that many MBs were skipped implies there is very little motion, and thus by merely counting the number of corresponding MBs that differ in their DC coefficients by some large enough value, a decision regarding the validity of the shot change can be made.

One way to improve this analysis would be to use any reference frame (I or P) that occurs immediately before the domain under consideration instead of using the previous I-frame. Since this reference frame occurs immediately before the domain, there is no room for noise between them to mislead the analysis. Of course, if the previous reference frame is a P-frame, then its DC coefficients need to be estimated from its residual error and the I-frame it was predicted from, for instance, using the technique in [13, 15].

3.3 Special Effects Detection

Apart from cuts or breaks, other types of edits between shots include fades, dissolves, wipes, etc. In a fade, the luminance gradually decreases to, or increases from, zero. In a dissolve, two shots, one increasing in intensity, and the other decreasing in intensity, are mixed. Wipes are generated by translating a line across the frame in some direction, where the content on the two sides of the line belong to the two shots separated by the edit. There are many other types of special effects, but most are quite uncommon.

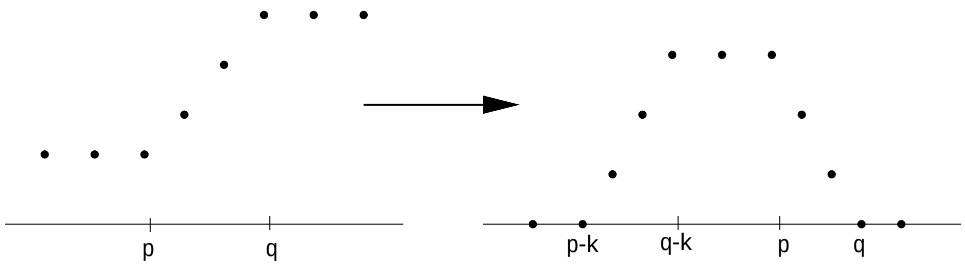


Figure 7: Dissolve detection method of Yeo and Liu.

These special effects cannot be detected with the use of MB types alone, since they occur gradually over a series of frames and the MBs tend to remain bidirectionally predicted or intra-coded over this span. Thus, it becomes necessary to use the DCT values of frames to perform this analysis. If the MPEG file contains P or B frames, the actual DCT values of their MBs need to be estimated.

Two techniques have been suggested by researchers. The paper by Yeo and Liu [16] suggests a method in which every frame is compared to the k^{th} frame following it. The separation parameter k should be larger than the number of frames in the edit. If that is the case, by using the sum of the absolute difference of the corresponding DC coefficients as the comparison metric, any ramp input should yield a symmetric plateau output with sloping sides, as shown in Figure 7.

Another technique suggested by Meng et al. [9] involves using the intensity variance to detect dissolves. They measured frame variance by using the DC coefficients of the I and P frames, and observed that during a dissolve the variance curve shows a parabolic shape.

One simple way to detect these edits would be to apply the DCT validation step to pairs of frames 5-10 apart in the sequence. We would see a significant decrease in performance and an expected increase in the number of false alarms.

We are currently working on detecting such transitions by clustering the frames after mapping each frame to a point in a low-dimensional space using a technique developed by Faloutsos and Lin [3] called *FastMap*, and observing the transitions between the resulting clusters. This technique is discussed in more detail in Section 7.

4 Motion Vector Analysis

Identifying shot breaks subdivides the entire clip into shots for which key frames need to be identified. If large camera motion is present in a single shot, then two frames that are spaced well apart in the shot may be quite dissimilar. In such cases, the shot is said to contain multiple scenes and more than one key frame may be required. Thus an initial step in performing such analysis is to identify consistent camera motion sequences as subshots, then to subdivide the shots into scenes which may yield multiple key frames. Earlier work on this problem can be found in [2, 8, 19].

4.1 Camera Motion Identification

Our approach involves using the motion vectors encoded into the MPEG format to determine any type of camera motion that may be present including zoom-in, zoom-out, pan left, pan right, tilt up, tilt down, or a combination of zoom, pan and tilt.[5]

The analysis for pan and tilt involves testing to see if a majority of the motion vectors are aligned in a particular direction. Each valid motion vector is compared with the unit vectors in the eight directions, and the number of motion vectors that lie in each of these directions is counted. If the direction receiving the highest number of vectors receives more than twice as many vectors as the second highest, a hypothesis is generated that the frame has camera motion in that direction.

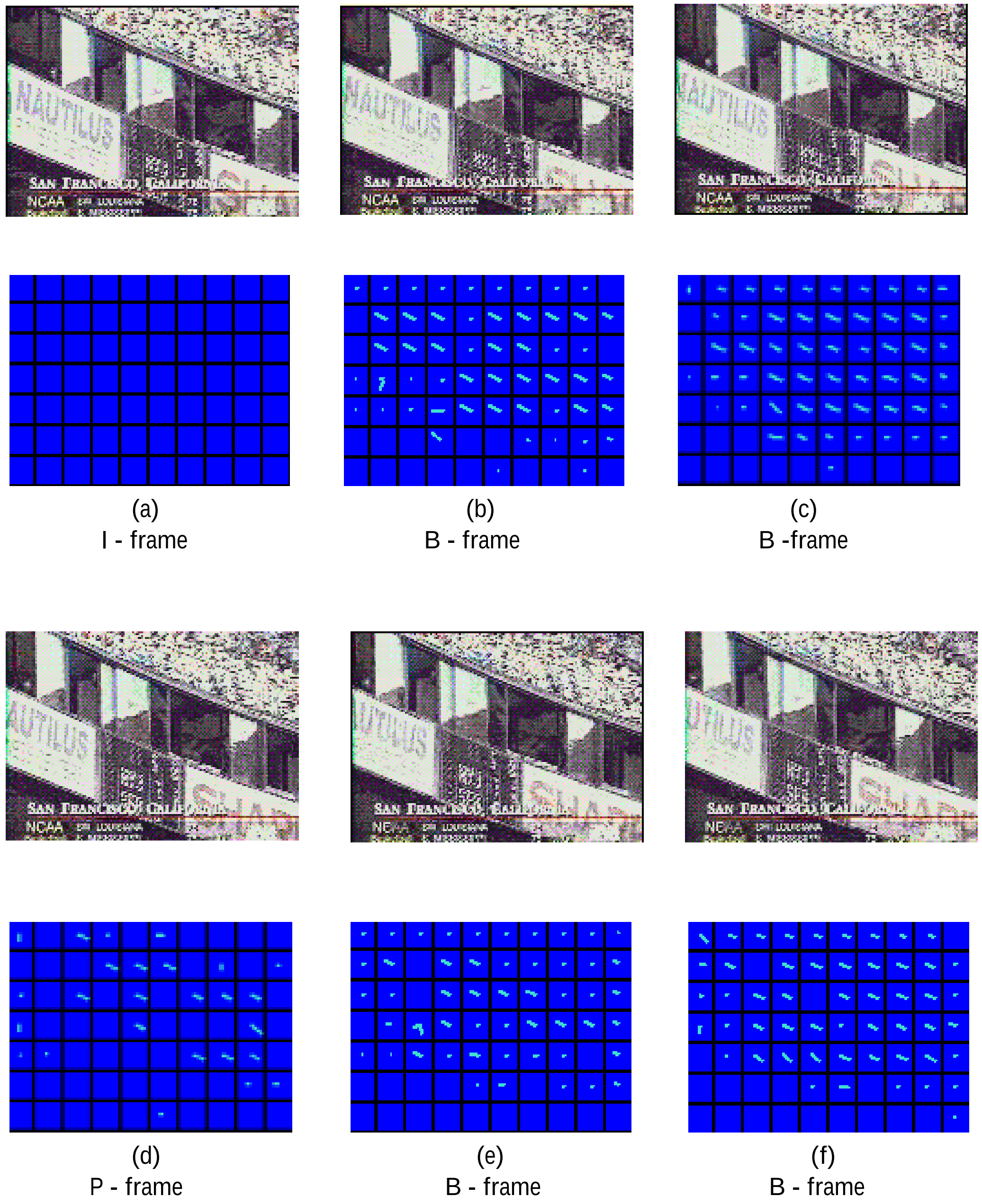
Figure 8 shows a pan and tilt combination sequence taken from a shot of the crowd at a football game, and the corresponding macroblock and motion vector display beneath each frame. In order to highlight the motion vectors, the MBs have not been distinguished using the colors shown in the legend in Figure 3(b). The macroblock display shows that most of the motion vectors are aligned with the direction of camera motion. Figure 9 shows a plot of the number of motion vectors aligned along each of the eight camera motion directions for the frame in Figure 8(b). The number of motion vectors aligned along each direction in the frame is proportional to the length of the ray along that direction in the plot.

A zoom model has been developed for testing zoom-in and zoom-out sequences. The zoom detector tests for the existence of a Focus of Expansion (FOE) or a Focus of Contraction (FOC) in each frame in a zoom sequence by using the motion vectors of each macroblock as flow data. A 2-D array of bins corresponding to the array of MBs is taken, and for each motion vector, a vote is cast for each bin lying along the path of a line segment along the motion vector. Thus, in the case of an FOC or an FOE, the bins in its vicinity would receive many votes. The detector also checks that the motion vectors near the FOE or FOC are small and that the average of their magnitudes over a constant radius around the FOC or FOE roughly increases with increasing radius.

Figure 10 shows a macroblock and motion vector display of a frame in a zoom sequence. As in the pan example, all the MBs have been shown with the same color. The arrangement of the motion vectors is as expected. These motion vectors are used by the zoom feature detector.

Two situations which cause problems: If there is a large moving object in the frame while the actual camera is stationary, or if a large moving object is being tracked by the camera. In the former case, the analysis might suggest that the camera is moving in the direction opposite to which the object is moving. In the latter case, it would suggest that the camera is stationary. For our purposes, we don't need to make these

[5] In our current work, we are assuming that the camera is stationary.



(a)
I - frame

(b)
B - frame

(c)
B -frame

(d)
P - frame

(e)
B - frame

(f)
B - frame

Figure 8: Pan example: six frames of a pan sequence and their motion vector arrangements.

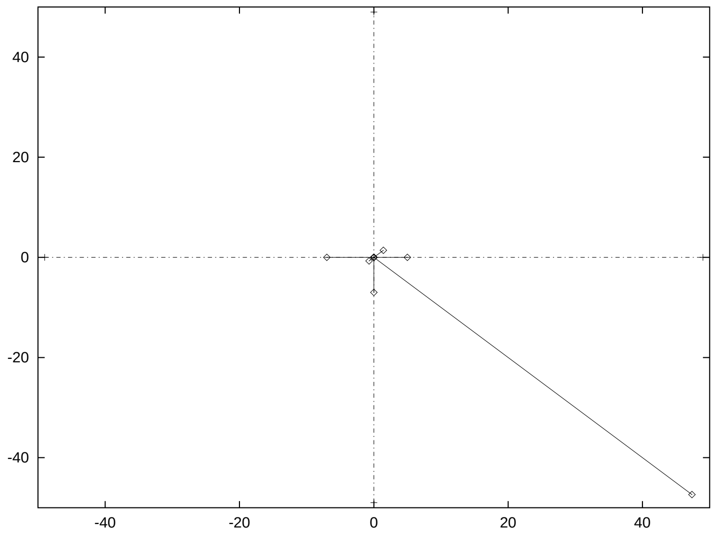


Figure 9: Pan example: Plot of the motion vector alignment density for the frame in Figure 8(b).

distinctions, as we are primarily interested in the dynamics of the content. Thus, if an object is being tracked, a scene change is not taking place, and if a large object is moving in the frame, then a scene change might be imminent. These observations are consistent with our goals.

A discussion of combinations of camera motions is appropriate at this juncture. Combinations of pans and tilts have already been considered. But what if a camera is panning and zooming-in at the same time? The FOE may not lie at the center of the frame; it might even lie outside the frame. Our zoom model checks for the presence of an FOE or FOC inside the frame only. If it does lie outside, the pan would be taken as the dominant motion; otherwise, the zoom would be the dominant motion.

5 Subshot Analysis

As mentioned earlier, due to large motions in a single shot, more than one key frame might be required to completely represent that shot. Questions that need to be answered include, "which frames should we select as additional key frames?" and, "how many additional key frames do we require?". We define a subshot as being a subset of consecutive frames in a shot that have some attribute in common. We consider two types of subshots, depending on which of two attributes, camera motion or shot content, the frames have in common: Motion subshots, and Content subshots. A motion subshot is a subshot that has a unique camera motion associated with it. A content subshot is a subshot whose frames have substantially similar content but differ in content from the adjacent subshots. As noted earlier, these content subshots are the different scenes found in the same shot.

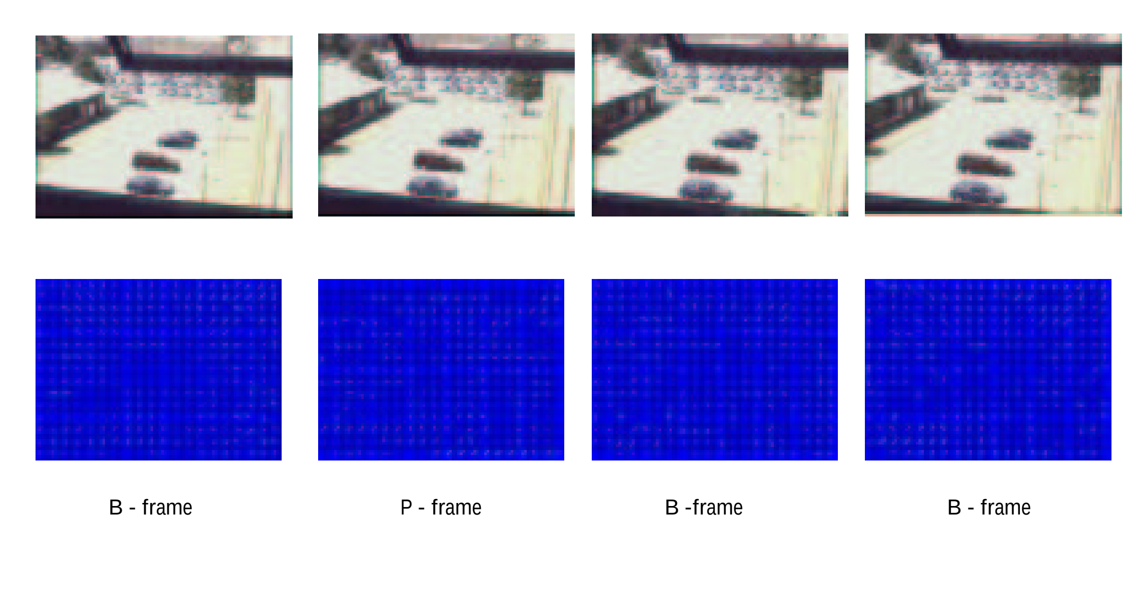


Figure 10: Zoom example : four frames of a zoom sequence and their motion vector arrangements.

5.1 Motion Subshots

After the camera motion detection phase, each frame is designated as belonging to one of the following eight translational motion classes — Pan-Right, Pan-Left, Tilt-Up, Tilt-Down, Pan-Right+Tilt-Up, Pan-Right+Tilt-Down, Pan-Left+Tilt-Up, Pan-Left+Tilt-Down; two zoom classes — Zoom-In, Zoom-Out; or two miscellaneous classes — No-Motion, Irregular-Motion. Sequences of frames in a shot that fall into the same valid motion class are grouped together into subshots, thus yielding a subdivision of the shots into subshots, according to these motion sequences. Short similar motion sequences that are close, but separated due to noise, are grouped to yield longer sequences.

As a direct consequence, detection of camera motion is used to increase the robustness of the macroblock-based video segmentation technique by using feedback to reject certain shot change candidates of low probability. Of course, a metric to record the strength of every shot change would be required for this to be feasible. For instance, if a shot change of low probability is detected at some frame, and the motion vector analysis shows that this frame lies inside a large zoom sequence, this detection can be invalidated. This will be part of our future work.

5.2 Content Subshots

Motion subshots have an obvious shortcoming: If there is a long pan or tilt sequence, the motion subshot will contain all the frames in that sequence, and will have only one key frame at its beginning. The sequence, however, might have captured many changes in content as it moved and all the necessary key frames would not be identified. This leads us to the subdivision of shots into content subshots with video content as the critical attribute.

Our approach still uses the motion information in shots to identify candidate key frames that might be different from the previously identified key frame. By comparing the DCT information of the candidate key frames with the previously identified key frame, we determine whether the candidate key frame is fact a true candidate for key frame extraction (Image Similarity).

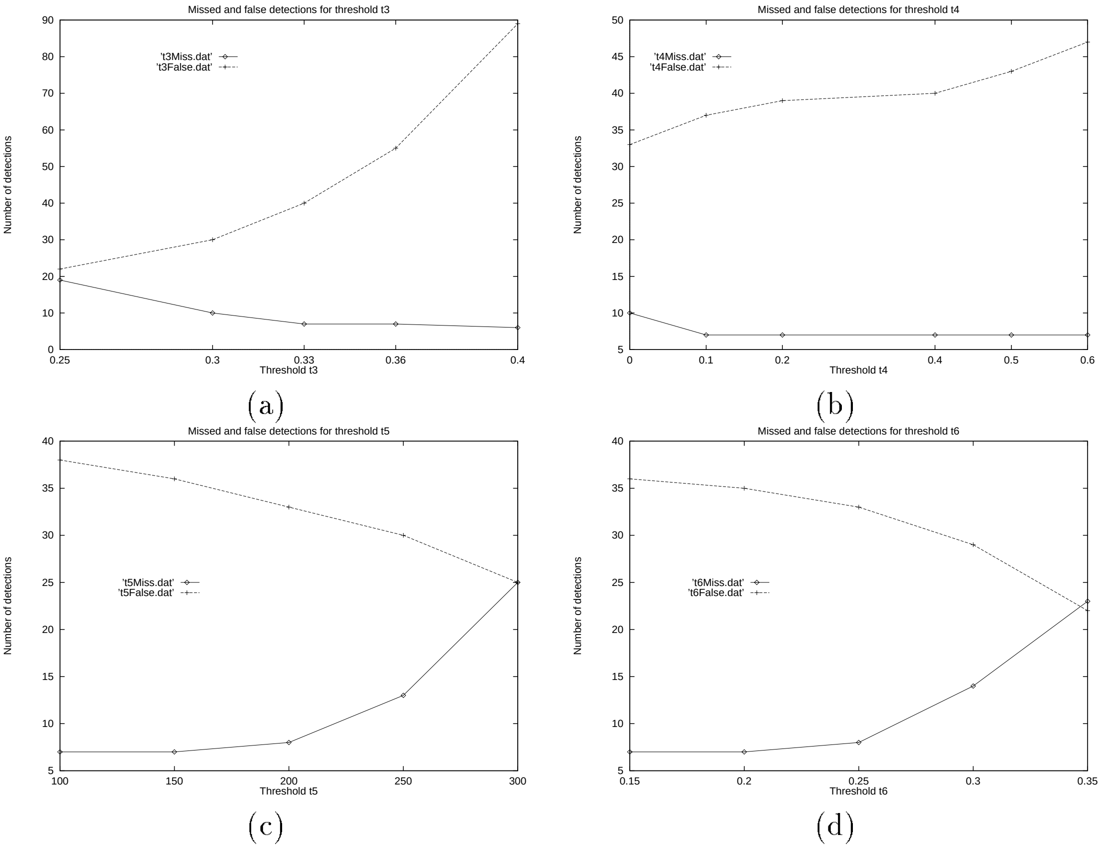
As mentioned earlier, each frame in a pan/tilt sequence has a particular direction associated with it. This procedure is extended by attaching the dominant or average magnitude of the motion vectors in that frame to it. By using vector addition and taking into consideration the fact these motion vectors are defined with respect to certain reference frames, a displacement vector is calculated for each frame with respect to the key frame identified at the beginning of this motion sequence. The magnitude of this displacement vector gives an estimate of the magnitude of the perceptual translation caused by the motion of the camera. By comparing this magnitude with the pixel dimensions of the frame, we determine whether the particular frame under consideration may differ in content from the key frame at the beginning of the sequence; if so, the frame is tagged as a candidate key frame.

5.3 Key Frame Identification

Key frames are selected as representatives for each of the subshots. Since key frames are the building blocks of any archival, indexing, or retrieval system, the best candidates for key frames are the I frames. Thus, the first I frame after the start of a new shot is chosen as the key frame. There are cases when the first available I frame could be considered to be too far into the shot. In such cases, a P frame can be chosen, and its DCT coefficients estimated using the technique described earlier [16].

6 Experiments and Results

Many tests used in this paper involve thresholds. Thresholds t_1, t_2, and t_3 were used in the segmentation process using MBs. Thresholds t_4, t_5, and t_6 were employed for the DCT validation process. The values for the thresholds given here were arrived at after trying many variations and combinations of values. Threshold t_1 was used for MPEG files containing only I and P frames, and since the test employing t_1 was the only test, a strong threshold of 0.33 was used. Higher values of t_1 increased the number of false detections. Threshold t_2 was used in a screening test (condition 1) for MPEG files with B frames also, and thus a conservative threshold of 0.5 was sufficient. Once the screening test was passed, t_3 with a value of 0.33 acted as the stricter threshold.



(a) (b)

(c) (d)

Figure 11: Segmentation results on video : Missed and false detections for (a) threshold t_3. (b) threshold t_4. (c) threshold t_5. (d) threshold t_6.

Threshold t_4 controlled which domains were routed to the DCT validation process, and thus was the most flexible. A value of between 0.4 to 0.5 was suitable. The lower the value of t_4, the better the results were, but on the other hand, the computation time increased. Threshold t_5 controlled the notion of disparity between MBs of consecutive I frames. A value of 200 was used for t_5. For threshold t_6, a value of 0.25 was used. Overall, these tests provided very lenient criteria for disparity between consecutive I frames for the segmentation process, and they proved sufficient. The thresholds used were global thresholds, but Meng et al. [9] suggested that an adaptive local window may perform better.

Figures 11(a) - (d) show the behavioral plots for thresholds t_3, t_4, t_5, and t_6. The 'miss' plots show the number of missed shot changes as a function of the threshold value, and the 'false' plots show the corresponding number of false detections. When some threshold was varied in those plots, other thresholds were held at the values suggested above, with the exception of threshold t_4 in the t_5 and t_6 plots, which was set at 0.0 to enable a complete study of the behavior of t_4.

For cut detection analysis, experiments were carried out on 61 different video clips

Table 1: Results of the segmentation process grouped by genre

Results by genre						
Genre	# Frames	# Actual	# Missed	# False	Recall %	Prec. %
animation	4137	50	1	4	98.0	92.4
commercial	2214	61	1	0	98.4	100.0
movie	6622	62	1	14	98.4	81.3
news	4089	51	0	3	100.0	94.4
sports	5674	62	4	16	93.5	78.4
surveillance	863	4	0	3	100.0	57.1
Total	23599	290	7	40	97.6	87.6

including football games, news clips, surveillance video clips, music videos, cartoons, and movies using a Sun SPARCstation 20 computer. The 23599 frames in those clips were digitized at frame rates varying from five frames per second (fps) to 30 fps. There were 290 shot changes.

The above-mentioned choices of the threshold values were made keeping in mind that the number of missed detections needs to be kept low. Once the number of missed detections seems to stabilize, the choice depends on which threshold value gives the fewest false detections.

Table 1 lists the number of missed and false detections for each genre of video clip. As expected, the sports clips gave the poorest results because they tend to contain large non-uniform object (player) motion and camera motion. The missed detections were due the fact that many shot changes have the same background (the playing field), so that the actual change is often very localized. Performance was best for the news clips, as they contain primarily scenes involving newscasters, and thus very little motion is involved. The other genres gave results that were not so extreme. The number of frames analyzed for the surveillance clips were probably not sufficient to warrant serious attention.

Recall can be defined as the fraction of actual shot changes correctly identified, while precision can be defined as the fraction of identified cuts that are correct. That is,

$$Recall = \frac{\#Actual - \#Missed}{\#Actual}$$

and

$$Precision = \frac{\#Actual - \#Missed}{\#Actual - \#Missed + \#False}$$

where $\#Actual - \#Missed$ is the number of the cuts identified by the algorithm.

The recall and precision values for the individual genres, as well as the total values, are also given in Table 1. With the suggested thresholds, there were 40 false detections

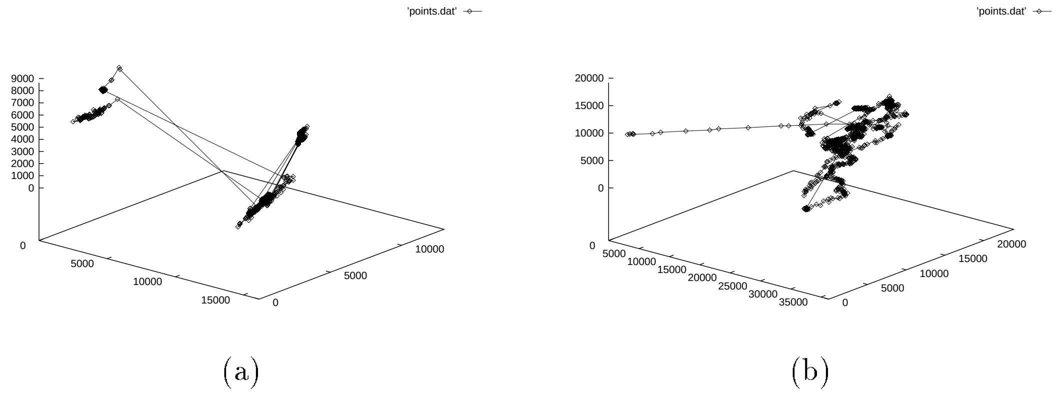


Figure 12: (a) Frame plot of a news interview clip. (b) Frame plot of a movie clip involving a fade-out at the end.

in the 23599 frames that were analyzed (0.17 %), and seven of the 290 actual shot changes were missed (2.4 %), resulting in a recall of 97.6 % and a precision of 87.6 %. More than half of the false detections occurred in a few football clips involving extreme close-ups of players in motion. A false detection is not as crucial as a missed detection, since the latter fails to detect an entire shot, whereas the former can at worst subdivide a single shot into many, thereby producing more than one key frame for the same scene.

The macroblock-based segmentation routine runs at speeds of up to 50 frames/sec for 160×120 MPEG video. Since the only processing involves extracting the basic information from the compressed representations, real-time speeds can easily be achieved using standard hardware.

A list of options implemented in the current version of our software is shown in the Appendix.

7 Discussion and Conclusions

We are currently developing techniques to detect special effect edits such as fades, dissolves, and wipes, using clusters of frames that represent shots and scenes and detecting transitions between these clusters. These transitions can be due to the camera, objects in the scene, or special effect edits. The transitions due to motion can be filtered out using the results obtained through motion vector analysis. The clusters are obtained by mapping each frame into a point in some low k-dimensional space. We use a technique called *FastMap* developed by Faloutsos and Lin [3], using the DC coefficients of all the frames and the Euclidean distance between the frames as the distance function to map the frames onto a point in the k-dimensional space.

Figure 12(a) shows an example of the clustering achieved in 3-dimensional space for the frames of a news clip involving an interview. The cuts are the sudden transitions from one cluster to another. Figure 12(b) shows an example of a fade-out at the end of a movie clip. The fade-out is represented by the string of points that protrude from the central sets of clusters. The advantage of this technique is that there need not be any restrictions on the types of the transitions or their behavior, such as linear variance of intensity during a fade, etc. On the other hand, the results of the motion vector analysis must be fairly robust to discard certain transitions as motion transitions.

We have presented a fast technique for detecting shot changes in MPEG compressed video without performing expensive decoding computations. Our technique uses the macroblocks of the P and B frames in MPEG video and uses the DCT information of I frames only for certain cases for which the MB information proves insufficient. The paper has also discussed techniques for subdividing shots based on various camera motions. Using these techniques, video can be parsed into shots and subshots and key frames from these shots can be used to perform various tasks like browsing, indexing, retrieval, etc. Our technique overcomes difficulties that other compressed-domain techniques might face; at the same time, it is fast compared to techniques that use DCT information throughout their analysis.

A Description of Software Options

```
usage: VideoSeg [ -dct | -valid | -full ]
                [ -dk ][ -dmv ][ -dv ][ -anno ][ -motion ]
                [ -writeDCT ][ -dumpScene ][ -dumpDesc ]
                [ -estimateDCT ][ -flow | -dispFlow ]
                [ -showKey ][ -dirpath PATH ]
                [ -help ] MPEGFilename

Note: Only one of the flags dct, valid, and full can be turned on.

The default method uses only the macroblock type information.
    -dct - uses the DC coefficients of I frames only
    -valid - performs DCT validation for certain detected cuts
             depending on a threshold
    -full - performs DCT validation for all detected cuts
The various thresholds explained in the text are currently hard-coded.

Options:

    -help  -  to display this
    -dk  -  to turn on khoros display of plots
    -dmv  -  to turn on Macroblock display of motion vectors
    -dv  -  to turn on display of motion vector directional votes
    -anno  -  to output annotation information
    -motion  -  to turn on motion analysis
    -writedct  -  to output dctinfo for key frames
    -dumpScene  -  to output scene and subscene information
    -dumpDesc  -  to output a description of the physical structure
                  of the video
    -estimateDCT  -  to output the DCT values of I,P,B frames
    -flow  -  to output the flow vectors
    -dispFlow  -  to display the flow vectors (does not output them)
    -showKey  -  to generate a montage of the identified key frames
    -dirpath PATH - PATH is where the various outputs will be stored

Note: Use one of -flow or -dispFlow, but not both !!
```

References

[1] F. Arman, A. Hsu, and M.Y. Chiu. Image processing on compressed data for large video databases. In *Proc. of the ACM Multimedia Conference*, pages 267–272, 1993.

[2] M. Cherfaoui and C. Bertin. Temporal segmentation of videos: A new approach. In *Proc. of the SPIE*, volume 2419, pages 38–47, 1995.

[3] C. Faloutsos and K. Lin. "FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets". In *Proc. of the ACM SIGMOD Conference*, 1995.

[4] A. Hampapur, R. Jain, and T. Weymouth. Digital video segmentation. In *Proc. of the ACM Multimedia Conference and Exposition*, pages 357–364, 1994.

[5] D. Le Gall. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34:46–58, 1991.

[6] D. Le Gall. The MPEG video compression algorithm: A review. In *Proc. of the SPIE*, volume 1452, pages 444–457, 1991.

[7] H.C. Liu and G.L. Zick. Scene decompostion of MPEG compressed video. In *Proc. of the SPIE*, volume 2419, pages 26–37, 1995.

[8] Junji Maeda. Method for extracting camera operations to describe sub-scenes in video sequences. In *Proc. of the SPIE*, volume 2187, pages 56–67, 1994.

[9] J. Meng, Y. Juan, and S.F. Chang. Scene change detection in a MPEG compressed video sequence. In *Proc. of the SPIE*, volume 2419, pages 14–25, 1995.

[10] A. Nagasaka and Y. Tanaka. Automatic video indexing and full-video search for object appearances. In *Proc. of the Working Conference on Visual Database Systems*, pages 119–133, 1991.

[11] I.K. Sethi and N. Patel. A statistical approach to scene change detection. In *Proc. of the SPIE*, volume 2420, pages 329–338, 1995.

[12] B. Shahraray. Scene change detection and content-based sampling of video. In *Proc. of the SPIE*, volume 2419, pages 2–13, 1995.

[13] K. Shen and J. Delp. A fast algorithm for video parsing using MPEG compressed sequences. In *Proc. of the IEEE International Conference on Image Processing*, pages 252–255, 1995.

[14] G.K. Wallace. The JPEG still image compression standard. *Communications of the ACM*, 34:30–44, 1991.

[15] B.L. Yeo and B. Liu. On the extraction of DC sequence from MPEG compressed video. In *Proc. of the IEEE International Conference on Image Processing*, pages 260–263, 1995.

[16] B.L. Yeo and B. Liu. Unified approach to temporal segmentation of motion JPEG and MPEG video. In *Proc. of the International Conference on Multimedia Computing and Systems*, pages 2–13, 1995.

[17] R. Zabih, J. Miller, and K. Mai. Feature-based algorithm for detecting and classifying scene breaks. In *Proc. of the ACM Multimedia Conference*, pages 189–200, 1995.

[18] H.J. Zhang, A. Kankanhalli, and S.W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems*, 1:10–28, 1993.

[19] H.J. Zhang, C.Y. Low, and S.W. Smoliar. Video parsing and browsing using compressed data. *Multimedia Tools and Applications*, 1:89–111, 1995.